In IEEE Symp. on Foundations of Comp. Sci. (FOCS'91). Copyright IEEE

A Linear Time Algorithm for Triconnectivity Augmentation[1] (Extended Abstract)

Tsan-sheng Hsu *Vijaya Ramachandran*
Department of Computer Sciences, University of Texas at Austin, Austin, TX 78712

Abstract

We consider the problem of finding a smallest set of edges whose addition triconnects an undirected graph. This is a fundamental graph-theoretic problem that has applications in designing reliable networks and fault-tolerant computing.

We present a linear time sequential algorithm for the problem. This is a substantial improvement over the best previous algorithm for this problem, which runs in $O(n(n+m)^2)$ time on a graph with n vertices and m edges.

1 Introduction

The problem of augmenting a graph to reach a certain connectivity requirement by adding edges has important applications in network reliability [5, 18] and fault-tolerant computing. One version of the augmentation problem is to augment the input graph to reach a given connectivity requirement by adding a smallest set of edges. We refer to this problem as the *smallest augmentation* problem.

The following results are known for solving the smallest augmentation problem on an undirected graph to satisfy a vertex connectivity requirement. Eswaran & Tarjan [2] gave a lower bound on the smallest number of edges for biconnectivity augmentation and proved that the lower bound can be achieved. Rosenthal & Goldner [16] developed a linear time sequential algorithm for finding a smallest augmentation to biconnect a graph; however, the algorithm in [16] contains an error. Hsu & Ramachandran [10] gave an corrected linear time sequential algorithm. An $O(\log^2 n)$ time parallel algorithm on an EREW PRAM using a linear number of processors for finding a smallest augmentation to biconnect an undirected graph was also given in Hsu & Ramachandran [10], where n is the number of vertices in the input graph. Watanabe & Nakamura [24, 26] gave an $O(n(n+m)^2)$ time sequential algorithm for finding a smallest augmentation to triconnect a graph with n vertices and m edges. There is no polynomial time algorithm known for finding a smallest augmentation to k-vertex-connect a graph, for $k > 3$.

Results on other versions of augmentation problems can be found in [1, 2, 4, 6, 7, 12, 14, 17, 21, 22, 23, 24, 25, 27].

In this paper, we present a linear time sequential algorithm for finding a smallest augmentation to triconnect a graph. The algorithm is divided into two stages. During the first stage, we biconnect the input graph. Then in the second stage, we triconnect the resulting biconnected graph using the smallest number of edges. We have to make sure that the total number of edges added in these two stages is minimum. It turns out that we cannot use the algorithm in Hsu & Ramachandran [10] to implement stage 1, since there exists a graph G such that any smallest augmentation for biconnecting G does not lead to a smallest augmentation for triconnecting G. An example is shown in Figure ??. Note that for edge-connectivity, it is shown in [14, 22] that there exists a smallest augmentation to k-edge-connect a graph G such that it is included in a smallest augmentation to $(k+1)$-edge-connect G, for an arbitrary k.

Owing to space limitation, many proofs are omitted in this abstract. They can be found in the full paper[11].

2 Definitions

<u>2-block</u> [2, 3, 16]
Given an undirected graph G with a set of vertices V, let $\mathcal{V} = \{V_i | 1 \le i \le k\}$ be a set of subsets of V such that $\cup_{i=1}^{k} V_i = V$ and two vertices u and w are in the same subset if and only if there is a simple cycle in G which contains u and w or $u = w$. The induced subgraph of G on each set of vertices V_i, $1 \le i \le k$, is a *2-block*. The union of all 2-blocks includes all edges in G that are not cut-edges. A 2-block containing only one vertex is called a *trivial 2-blocks*. A trivial *2-block*

[1] This work was supported in part by NSF Grant CCR-89-10707.

of G is called a *cut-block* if it is a cutpoint.

2-block graph

Given an undirected graph G, we define its *2-block graph*, 2-$blk(G)$, as follows. Each cutpoint and 2-block that is not a cut-block is represented by a vertex in $blk(G)$. The vertices of 2-$blk(G)$ that represent blocks that are not 2-blocks are called *b-vertices* and those representing cutpoints are called *c-vertices*. For a vertex u in 2-$blk(G)$, let $V_u = \{u\}$ if u is a c-vertex and $V_u = \{w | w$ is a vertex in the corresponding 2-block represented by $u\}$ if u is a b-vertex. The induced subgraph of G on V_u is the *corresponding subgraph* of u. Two vertices u and w in 2-$blk(G)$ are adjacent if and only if any one of the following conditions is true: (i) $|V_u| = 1$, $|V_w| = 1$ and the vertex in V_u and the vertex in V_w are adjacent in G; (ii) $|V_u| = 1$ and $V_u \subset V_w$; (iii) $|V_w| = 1$ and $V_w \subset V_u$. It is well-known that 2-$blk(G)$ is a forest. If G is connected, 2-$blk(G)$ is a tree. If 2-$blk(G)$ is a tree, we refer to it as a *2-block tree*. For a vertex v in G, let $d_2(v)$ be the degree of its corresponding c-vertex in 2-$blk(G)$ if v is a cutpoint. If v is not a cutpoint, let $d_2(v) = 1$. For more on properties of 2-$blk(G)$, see [10]. An example of a graph and its 2-$blk(G)$ is shown in Figure ??.

Tutte component

The *Tutte components* of a biconnected graph are the triconnected components defined in Tutte [20] (see also Ramachandran [15]). Each Tutte component can be a single vertex, a triconnected component, a simple cycle (a polygon) or a pair of vertices with at least three edges between them (a bond).

3-block graph

Given a 2-block H of G, we define the *3-block graph*, 3-$blk(H)$, as follows. The 3-block graph contains three sets of vertices: β-vertices, σ-vertices and π-vertices. For every Tutte component of H that is a single vertex or a triconnected component, we create a β-vertex in 3-$blk(H)$. A Tutte component that consists of only one vertex is a *trivial Tutte component*. For every polygon Q of H, we create a π-vertex; if w is a vertex in Q with degree 2 in H, we create a β-vertex b_w for w, and we call w *the corresponding Tutte component represented by* b_w. Let z_1 and z_2 be the two vertices in Q that are adjacent to w. We create a σ-vertex for the pair of vertices z_1 and z_2. For every pair of vertices a_1 and a_2 in H, we create a σ-vertex for (a_1,a_2) if we have performed a Tutte split with respect to a_1 and a_2 in H. An example of a σ-vertex, β-vertex and π-vertex is shown in Figure ??.

Vertices u and v in 3-$blk(H)$ are adjacent if any one of the following conditions is true: (i) u is a β-vertex corresponding to a Tutte component H_u that is a triconnected component, v is a σ-vertex and H_u contains the pair of vertices corresponding to v; (ii) u is a β-vertex corresponding to a degree-2 vertex w in H and v is the σ-vertex corresponding to the pair of vertices in H that are adjacent to w; (iii) u is a π-vertex corresponding to a polygon Q in H, v is σ-vertex corresponding to a pair of vertices z_1 and z_2 in Q; (iv) interchanging u and v in any one of the previous three conditions. We call the pairs of vertices that correspond to σ-vertices *Tutte pairs*. It is known that the number of Tutte pairs in an n-node biconnected graph is $O(n)$ [13, 15].

¿From [9, 15, 20], we know that 3-$blk(H)$ is a tree if H is a 2-block. We call this tree a *3-block tree*. We call the set of trees corresponding to 2-blocks in G *the 3-block graph* of G or 3-$blk(G)$. Each Tutte component that corresponds to a β-vertex in the 3-block graph is a *3-block* of G. Given a graph G with n vertices and m edges, 2-$blk(G)$ and 3-$blk(G)$ can be computed in $O(n+m)$ time using procedures in [9, 15]. Examples of 3-block graphs are shown in Figures ?? & ??.

The implied path in the 2-block graph

Let G be an undirected graph and let u and v be two distinct vertices in G. Let G' be the graph obtained from G by adding the edge (u,v) and let b_u and b_v be the two vertices in 2-$blk(G)$ whose corresponding subgraphs contain u and v, respectively. We denote the path P_2 between b_u and b_v in 2-$blk(G)$ *the implied path between u and v in 2-$blk(G)$* (we let $P_2 = [b_u]$ if there is no such path).

Block-sequence, separating-sequence and cut-sequence

Let Y be the set of all b-vertices in P_2 and those c-vertices in P_2 whose corresponding cutpoints are cut-blocks. Let $|Y| = r$ and y_i be the ith vertex in Y encountered when we traverse P_2 starting form b_u to b_v. We construct a *separating-sequence* $W = [w_1 = u, w_2, \cdots, w_{2r-1}, w_{2r} = v]$ for vertices u and v such that for $1 < i < r$, w_{2i-1} and w_{2i} are the two cutpoints adjacent to y_i if y_i is a b-vertex; w_{2i-1} and w_{2i} are the two vertices adjacent to y_i in G if y_i is a cutpoint; w_2 and w_{2k-1} are cutpoints adjacent to y_1 and y_r, respectively. Note that w_{2i} and w_{2i+1} can be equal, for $1 \leq i < r$; they are different if and only if any there is at least one c-vertex in $\{y_i,y_{i+1}\}$. A *cut-sequence* $C = [u, c_1, \cdots, c_x, v]$ for vertices u and v is a sequence of vertices in G such that c_i is the cutpoint corresponding to the ith c-vertex encountered when we traverse P_2 from b_u to b_v and x is the number of c-vertices in P_2. An example is shown in Figure ??.

The collection of implied paths

For $1 \leq i \leq r$, let H_i be the corresponding 2-block represented by y_i if y_i is b-vertex; let H_i be the corresponding cutpoint represented by y_i if y_i is a c-vertex. For $1 \le i \le r$, let $Q_i = [H_i]$ if y_i is a c-vertex; otherwise, let Q_i be the path in 3-$blk(H_i)$ between the two β-vertices that correspond to Tutte components containing w_{2i-1} and w_{2i}. Each Q_i, $1 \le i \le r$, is chosen such that both w_{2i-1} and w_{2i} are each contained in exactly one of the Tutte components corresponding to β-vertices in Q_i if y_i is a b-vertex. Note that w_{2i} and w_{2i+1} can be equal, for $1 \le i < r$; they are different if and only if there are a sequence of more than one cutpoint between y_i and y_{i+1} in P_2. If there is more than one Tutte component that contains w_i, $1 \le i \le 2r$, we choose an arbitrary one. We call $\mathcal{Q} = \{Q_1, \cdots, Q_r\}$ *the collection of implied paths between u and v in 3-*$blk(G)$. If G is biconnected, $r = 1$; Q_1 is also called *the implied path between u and v in 3-*$blk(G)$. An example of these definitions is shown in Figure ??.

Separating degree

Given a σ-vertex s, let (a_1,a_2) be its corresponding Tutte pair. We define $d_3((a_1,a_2))$ or $d_3(s)$ to be the degree of z in the 3-block graph. We know that $d_3((a_1,a_2)) \ge 2$ for any Tutte pair (a_1,a_2). Recall that for a vertex v in G, $d_2(v)$ is the degree of v in 2-$blk(G)$ if v is a cutpoint; $d_2(v)$ is 1 if v is not a cutpoint. Given a graph with h connected components, the *separating degree* of a Tutte pair (a_1,a_2), $sd((a_1,a_2))$, is $\sum_{i=1}^{2} d_2(a_i) + d_3((a_1,a_2)) + h - 4$. We will show (Claim 1 in Section 3) that the separating degree of a Tutte pair is equal to the smallest number of edges needed to connect the graph obtained from G by removing the Tutte pair. The separating degree for the corresponding σ-vertex s, $sd(s)$, is equal to $sd((a_1,a_2))$.

The triconnectivity augmentation number

Given a graph G, the *triconnectivity augmentation number* of G is the smallest number of edges needed to triconnect G.

3 A Lower Bound for the Triconnectivity Augmentation Number

In this section, we state a lower bound for the triconnectivity augmentation number. This lower bound turns out to be the exact triconnectivity augmentation number.

Separating degrees of σ-vertices

We state a claim to justify the way the separating degree of a Tutte pair in a 3-block graph is defined.

Claim 1 *The separating degree of a Tutte pair is equal to the smallest number of edges needed to connect the graph obtained from G by removing the Tutte pair.* □

3-block leaf and isolated 3-block vertex

We identify β-vertices in a 3-block graph whose corresponding Tutte components must contain a new incoming edge if we want to triconnect the input graph.

Definition 1 *Given a β-vertex b, let H_b be its corresponding Tutte component of G. A degree-1 vertex b is a* **3-block leaf** *if any one of the following conditions is true: (i) H_b consists of only one vertex u and u is not a cutpoint; (ii) if H_b contains any cutpoint c of G, c is in a Tutte pair of G contained in H_b. (Note that (i) holds if and only if u is in a polygon.) A degree-0 β-vertex b is an* **isolated 3-block vertex** *if any one of the following conditions is true: (i) H_b contains at most 2 cutpoints; (ii) H_b consists of only one vertex u and the degree of u in G is at most 2.*

Figure ?? illustrates a 3-block leaves. Note that if G is biconnected, then a degree-1 vertex in 3-$blk(G)$ must be a 3-block leaf.

Demanding vertex

Definition 2 *Given a 3-block leaf or an isolated 3-block vertex b in 3-blk(G), let H_b be its corresponding Tutte component. A vertex u in H_b is a* demanding vertex *of b if any one of the following conditions is true: (i) u is not a cutpoint or in any Tutte pair of G contained in H_b; (ii) b is an isolated 3-block vertex and H_b consists of only the vertex u. The vertex u is also called a* demanding vertex *in G.*

Claim 2 *Let b be a 3-block leaf or an isolated 3-block vertex in 3-blk(G). Then there is at least one demanding vertex of b.* □

When we specify the 3-block graph of a graph G, we will include a demanding vertex for each 3-block leaf and each isolated 3-block vertex in 3-$blk(G)$.

The weight of a graph

We now define the *weight* of a graph, which we will relate later to the number of edges needed to triconnect the graph.

Definition 3 *Let H be a 2-block in a graph G whose corresponding vertex in 2-blk(G) is r_H. Let $l_3(H)$ be the number of 3-block leaves in H. We define* the weight of the graph *G, $w(G)$, to be $\sum_{\forall \text{ 2-blocks } H} max\{3 - d_2(r_H), l_3(H)\}$, if G is not biconnected. Otherwise, let $w(G) = l_3(G)$.*

Massive, critical and balanced

For an undirected graph G with the weight $w(G)$, a Tutte pair z or its corresponding σ-vertex is *massive* if $sd(z) > \lceil \frac{w(G)}{2} \rceil$. A Tutte pair z or its corresponding σ-vertex is *critical* if $sd(z) = \lceil \frac{w(G)}{2} \rceil$. If no Tutte pair in G is *massive*, then G and its 3-block graph are called *balanced.* Given a σ-vertex v in 3-$blk(G)$, let 3-$blk(G) - v$ be the graph obtained from 3-$blk(G)$ by removing v. A *v-chain* is a component of 3-$blk(G) - v$ which contains only one 3-block leaf in 3-$blk(G)$. The 3-block leaf of 3-$blk(G)$ in a v-chain is called the *v-chain leaf.*

A lower bound of the triconnectivity augmentation number

We now state a lower bound for the triconnectivity augmentation number for a general undirected graph.

Lemma 1 *Given an undirected graph G, we need at least $max\{d, \lceil \frac{w(G)}{2} \rceil\}$ edges to triconnect G, where d is the largest separating degree among all σ-vertices in 3-$blk(G)$.*

Proof: The first component of the lower bound comes from Claim 1. For the other component of the lower bound, suppose that G' is triconnected and is obtained from G by adding a smallest set of edges. For each 2-block H, we must have at least $3 - d_2(r_H)$ new incoming edges in G'. For each 3-block leaf in 3-$blk(H)$, the induced subgraph on vertices corresponding to it must contain at least one new incoming edge in G'. Hence we need at least $\lceil \frac{max\{3-d_2(r_H), l_3(H)\}}{2} \rceil$ new edges for each 2-block H. The total number of new edges needed is thus $\lceil \frac{w(G)}{2} \rceil$. Hence we need at least $max\{d, \lceil \frac{w(G)}{2} \rceil\}$ edges in G'. This proves the lemma. $\Box$

4 Finding a Smallest Augmentation to Triconnect a Biconnected Graph

In this section, we consider the problem of finding a smallest set of edges to triconnect a biconnected graph. We show that the lower bound given in Lemma 1 can be achieved, and we give a linear time algorithm to find a smallest augmentation to triconnect the graph.

4.1 Properties of the 3-Block Tree for a Biconnected Graph

In this section, we explore properties of 3-$blk(G)$ that will be used in the following sections, where G is a biconnected graph.

First we give bounds on the number of massive and critical σ-vertices in 3-$blk(G)$. They are similar to the bounds on the number of *massive* and *critical c-vertices* in 2-$blk(G)$ given in [10]. Proofs in [10] can be easily modified to prove the following claim.

Claim 3 *Let s_1 be a σ-vertex with the largest separating degree among all σ-vertices in 3-$blk(G)$ and let s_i be a σ-vertex with the largest separating degree among all σ-vertices other than $s_1, \cdots, s_{i-1}$, for $2 \le i \le 3$. Then the following five properties hold.*
(i) $\sum_{i=1}^{3} d_3(s_i) - 4 \le l_3(G)$*, where $l_3(G)$ is the number of degree-1 vertices in 3-$blk(G)$.*
(ii) If $blk(G)$ has more than two σ-vertices, then $d_3(s_3) \le \frac{l_3(G)+4}{3}$.
(iii) There can be at most one massive σ-vertex in $blk(G)$.
(iv) If there is a massive σ-vertex in $blk(G)$, then there is no critical σ-vertex in $blk(G)$.
(v) There can be at most two critical σ-vertices in $blk(G)$ if $l_3(G) > 2$. $\Box$

Given an input biconnected graph G, its 3-block graph and a graph G' obtained from G by adding an edge between two distinct vertices u and v, we now describe a method to obtain 3-$blk(G')$ from 3-$blk(G)$ by local updating operations on 3-$blk(G)$ instead of computing it directly from G'. Let P_3 be the implied path between u and v in 3-$blk(G)$.

We define the *implied graph G_{P_3} of G on the path* P_3 as follows. G_{P_3} contains vertices and edges that are in Tutte components of G corresponding to β-vertices in P_3. For every π-vertex q in P_3, let s_a^q and s_b^q be the two σ-vertices connected to q in P_3. Let (a_1^q, a_2^q) and (b_1^q, b_2^q) be the Tutte pairs represented by s_a^q and s_b^q, respectively. If s_a^q is adjacent to a β-vertex in P_3 that corresponds to a trivial Tutte component $\{w\}$, then let $a_1^q = w$ and let $a_2^q = w$. The same procedure is applied on s_b^q, b_1^q and b_2^q. We assume that if we traverse clockwise around the simple cycle C_q represented by q starting from a_1^q, the sequence of vertices visited in $\{a_2^q, b_1^q, b_2^q\}$ is b_1^q, b_2^q and a_2^q. We add the edge (a_1^q, b_1^q) to G_{P_3} if $a_1^q \ne b_1^q$ and we add the edge (a_2^q, b_2^q) to G_{P_3} if $a_2^q \ne b_2^q$. An example is shown in Figure **??**.

Claim 4 *Given a biconnected graph G and two demanding vertices u and v in G where u and v are in different Tutte components, let P_3 be the implied path in 3-$blk(G)$ between u and v. The implied graph on P_3, G_{P_3}, is biconnected if and only if u and v are not both in trivial Tutte components or there is more than one π-vertex in P_3.* $\Box$

We now define the *crack* operation which is useful in describing the relation between 3-$blk(G)$ and the 3-block graph of G after adding an edge.

Definition 4 *Let G be a biconnected graph and let G' be the graph obtained from G by adding an edge between two demanding vertices u and v in G, where u and v are in different Tutte components. Let $P_3, q, s_a^q, s_b^q, a_1^q, a_2^q, b_1^q$ and b_2^q be the path and vertices defined above. The* **crack** *operation on q with respect to P_3 consists of the following procedures. (i) We add the edge (a_1^q, b_1^q) to the simple cycle C_q corresponding*

to q if $a_1^q \neq b_1^q$ and (a_1^q,b_1^q) is not an edge in C_q. The edge (a_2^q,b_2^q) is added under the same condition for a_2^q and b_2^q. For each new simple cycle C created by adding these edges, we create a new π-vertex if C contains a Tutte pair in C_q (excluding (a_1^q,a_2^q) and (b_1^q,b_2^q)).
(ii) After performing operations given in part i, let q_1 and q_2 be the two π-vertices corresponding to simple cycles created in part i that contain only (a_1^q,b_1^q) and only (a_2^q,b_2^q), respectively. New σ-vertices s_1 (corresponds to new Tutte pair (a_1^q,b_1^q)) and s_2 (corresponds to new Tutte pair (a_2^q,b_2^q)) are added and connected to q_1 and q_2, respectively. (iii) After performing operations given in parts i and ii, let s be a σ-vertex in 3-$blk(G)$ such that $s \neq s_a^q$ and $s \neq s_b^q$. An edge (s,q) in 3-$blk(G)$ is changed to (s,q_1) or (s,q_2) depending on whether the Tutte pair represented by s is on the simple cycle represented by q_1 or by q_2.

Note that the *crack* operation creates at most two π-vertices. If there are two π-vertices created by performing the *crack* operation on a π-vertex q with respect to a path P_3, then P_3 is *non-adjacent* on q. Otherwise, P_3 is *adjacent* on q. An example is shown in Figure ??.

Lemma 2 states the relations between 3-$blk(G)$ and 3-$blk(G')$, where G' is the graph obtained from G by adding an edge.

Lemma 2 *Let G be an input biconnected graph and let G' be the graph obtained from G by adding an edge between two vertices u and v in G, where u and v are in different Tutte components. Let P_3 be the implied path between u and v in 3-$blk(G)$. We can obtain 3-$blk(G')$ from 3-$blk(G)$ by applying the following operations. (i) Edges in P_3 are eliminated. Vertices and edges in 3-$blk(G)$ that are not in P_3 remain in 3-$blk(G')$. (ii) The β-vertices in P_3 shrink into a new β-vertex b_{P_3} in 3-$blk(G')$ if u and v are not both in trivial Tutte components or there is more than one π-vertex in P_3. The corresponding Tutte component of b_{P_3} is $G_{P_3} \cup (u,v)$, where G_{P_3} is the implied graph on P_3. Otherwise, β-vertices in P_3 are eliminated. (iii) A vertex s in P_3 with $d_3(s) = 2$ is eliminated in 3-$blk(G')$ if s is a σ-vertex or a π-vertex. A σ-vertex s in P_3 with $d_3(s) \geq 3$ is adjacent to b_{P_3} if b_{P_3} exists. (iv) A π-vertex q in P_3 with $d_3(q) \geq 3$ is* **cracked** *(Definition 4) and new π-vertices are connected to b_{P_3} if b_{P_3} exists. If b_{P_3} does not exist, we create a new π-vertex q' and the two σ-vertices created by the* **crack** *operation are connected to q'.*

Proof: Parts i and iii are trivial. We only prove parts ii and iv. G_{P_3} is biconnected (Claim 4) if u and v are not both in trivial Tutte components or there is more than one π-vertex in P_3. ¿From the definition of G_{P_3}, we know that the removal of any Tutte pair in G_{P_3} will result in a graph with two connected components which contain u and v, respectively. Thus the graph G'_{P_3} obtained from G_{P_3} by adding (u,v) is triconnected. It is also true that G'_{P_3} is maximal (no vertex can be added such that the resulting graph is still triconnected). This proves part ii. We know that pairs of endpoints between edges added in cracking π-vertices correspond to new Tutte pairs. Every new Tutte pair created is contained in G'_{P_3}. This proves part iv. □

An example of updating the 3-block graph is shown in Figure ??. The following is a corollary of Lemma 2.

Corollary 1 *Let G' be the graph obtained from G by adding an edge between any two demanding vertices u and v in G. The degree of a σ-vertex s in P_3, the implied path in 3-$blk(G)$ between u and v, with degree > 2 in 3-$blk(G)$ is decreased by 1 in 3-$blk(G')$.* □

Definition 5
[The leaf-connecting condition for triconnecting a biconnected graph]
Let G be a biconnected graph and let u and v be two demanding vertices in G. Let P_3 be the implied path between u and v in 3-$blk(G)$. The pair of vertices u and v satisfies the **leaf-connecting condition** *if and only if any one of the following conditions is true: (i) the path P_3 contains a β-vertex of degree at least 4; (ii) the path P_3 contains two vertices of degree at least 3; (iii) there exists a π-vertex in P_3 such that P_3 is* non-adjacent *on it.*

Lemma 3 *Let G be a biconnected graph and G' be the graph obtained from G by adding a new edge (u,v). Let $l_3(G)$ and $l_3(G')$ be the numbers of degree-1 vertices in 3-$blk(G)$ and 3-$blk(G')$, respectively. If u and v satisfy the leaf-connecting condition (Definition 5) and $l_3(G) > 3$, then $l_3(G') = l_3(G) - 2$.*

Proof: We know that two degree-1 vertices in 3-$blk(G)$ that correspond to Tutte components containing u and v are eliminated. If u and v satisfy the leaf-connecting condition, the only β-vertex created has a degree at least 2. Since we do not create any degree-1 vertex, the lemma holds. □

4.2 An Algorithm for Triconnecting a Biconnected Graph Using the Smallest Number of Edges

We will show after the description of the algorithm that by using algorithm aug2to3 given below, the lower bound given in Lemma 1 can be reduced by 1 each time we add a new edge. We now describe the algorithm.

{∗ The algorithm finds a smallest augmentation to triconnect G, where G is biconnected with ≥ 4 vertices. ∗}
graph function aug2to3(**graph** G); {∗ The algorithmic notation used here is from Tarjan [19]. ∗}
$T := 3\text{-}blk(G)$;
let $l_3(G)$ be the number of degree-1 vertices in T;
do $l_3(G) \geq 2 \rightarrow$
let s_1 be a σ-vertex with the largest degree in T;
1. **if** s_1 is massive $\rightarrow v := s_1;\ w := s_1$
| s_1 is not massive $\rightarrow$
let s_2 be a σ-vertex with the largest degree in T other than s_1;
let $d_3(s_2) = 0$ if s_2 does not exist;
let b_1 be a non-σ-vertex with the largest degree in T;
if $d_3(s_1) > 2 \rightarrow$
2. $v := s_1$;
3. **if** $d_3(s_2) > 2 \rightarrow w := s_2$
| $d_3(s_2) \leq 2$ **and** $d_3(b_1) > 2 \rightarrow w := b_1$
| $d_3(s_2) \leq 2$ **and** $d_3(b_1) \leq 2 \rightarrow w := v$
fi
| $d_3(s_1) \leq 2 \rightarrow$
let b_2 be a non-σ-vertex with the largest degree in T other than b_1;
4. $v := b_1$;
if $d_3(b_2) > 2 \rightarrow v := b_2$
| $d_3(b_2) \leq 2 \rightarrow w := v$ **fi**
fi
fi;
if $v = w$ **and** v is a π-vertex **and** $d_3(v) > 3 \rightarrow$
5. find two degree-1 vertices y and z such that the path between them in T is non-adjacent on v
| $v \neq w$ **or** v is not a π-vertex **or** $d_3(v) \leq 3 \rightarrow$
6. **if** s_1 is massive $\rightarrow$ find two s_1-chain leaves y and z
| s_1 is not massive $\rightarrow$
find two degree-1 vertices y and z such that the path between them passes through v and w
fi
fi;
find demanding vertices u_1 and u_2 of y and z, respectively; {∗ Claim 2 shows that this is always possible. ∗}
add an edge between u_1 and u_2;
update the 3-block graph T;
if $l_3(G) \neq 3 \rightarrow l_3(G) := l_3(G) - 2$
| $l_3(G) = 3 \rightarrow l_3(G) := l_3(G) - 1$ **fi**
od;
return G
end aug2to3;

Before the proof of correctness for algorithm aug2to3, we state a claim for an input graph G that is unbalanced. The proof of this claim is similar to a proof in [10, 16].

Claim 5 *Let G be an unbalanced biconnected graph with at least 4 vertices. Let s be the massive σ-vertex in 3-$blk(G)$ and let $\delta = d_3(s) - 1 - \lceil \frac{l_3(G)}{2} \rceil$. There are $2\delta + 2$ s-chains in 3-$blk(G)$. Let $\mathcal{M}$ be the set of s-chain leaves. By adding $2k$, $k \leq \delta$, edges to connect $2k+1$ vertices of $\mathcal{M}$, we reduce both $d_3(s)$ and the number of leaves in the 3-block tree by k.* □

Claim 6 *Let d be the largest separating degree among all σ-vertices in 3-$blk(G)$ and let $l_3(G)$ be the number of degree-1 vertices in 3-$blk(G)$. If G is biconnected with at least 4 vertices, we can triconnect G by adding $max\{d, \lceil \frac{l_3(G)}{2} \rceil\}$ edges using algorithm aug2to3.*

Proof: If 3-$blk(G)$ contains a massive vertex s_1, then the two s_1-chain leaves are found in steps 1 and 6. If 3-$blk(G)$ is balanced, all critical vertices are found in steps 2 and 3. Thus if $d \geq \lceil \frac{l_3(G)}{2} \rceil$, then d is decreased by 1 (Corollary 1). The pair of vertices u_1 and u_2 also satisfies part *ii* of the leaf-connecting condition (Definition 5) by steps 2, 3 and 4 if possible. Otherwise, it satisfies part *i* of the leaf-connecting condition by step 4 or part *iii* of the leaf-connecting condition by steps 4 and 5. Thus if $d < \lceil \frac{l_3(G)}{2} \rceil$, $l_3(G)$ decreases by 2 each time algorithm aug2to3 adds an edge. The algorithm guarantees that $max\{d, \lceil \frac{l_3(G)}{2} \rceil\}$ decreases by 1 each time we add an edge. We know that $max\{d, \lceil \frac{l_3(G)}{2} \rceil\} = 0$ implies that 3-$blk(G)$ is a single β-vertex. Thus G is triconnected if G contains at least 4 vertices. Hence the claim is true. □

Theorem 1 *The triconnectivity augmentation number for a biconnected graph G equals $max\{d, \lceil \frac{l_3(G)}{2} \rceil\}$, where $d+1$ is the largest degree among all σ-vertices in 3-$blk(G)$ and $l_3(G)$ is the number of degree-1 vertices in 3-$blk(G)$.* □

Claim 7 *Algorithm aug2to3 runs in $O(n+m)$ time on a graph with n vertices and m edges.* □

5 Finding a Smallest Augmentation to Triconnect a Graph

In this section, we consider the problem of finding a smallest augmentation to triconnect any undirected graph. We show that the lower bound given in Lemma 1 can be always achieved by giving an algorithm for it. Finally, we describe a linear time implementation for the algorithm.

5.1 Properties of the 3-Block Graph

In this section, we study properties of the 3-block graph that are used in the next section for designing an algorithm to find a smallest augmentation to triconnect a graph. We first study the changes made on the 3-block graph after adding an edge into the original graph in Definition 6 and Lemma 4. Then in Claim 8, Claim 9, Lemma 5 and Corollary 3, we show that we can always decrease the lower bound given in Lemma 1 by 1 by adding an edge.

Let G' be the graph obtained from G by adding an edge between two demanding vertices u and v in G. We describe the updating operation performed on 3-$blk(G)$ after adding an edge.

Definition 6 *Let G' be the graph obtained from G by adding an edge between two demanding vertices u and v in G, where u and v are in different Tutte components. Given $\{Q_1, \cdots, Q_r\}$, the collection of implied paths in 3-$blk(G)$ between u and v, the block sequence $[y_1, \cdots, y_r]$, the separating-sequence $[w_1, \cdots, w_{2r}]$ and the cut-sequence $[u, c_1, \cdots, c_x, v]$, we can obtain 3-$blk(G')$ from 3-$blk(G)$ by performing the following operations. (i) For each Q_i, $1 \leq i \leq r$, we perform the operation given in Lemma 2 on the 3-block tree that contains Q_i if Q_i contains more than 1 vertex. (ii) A new π-vertex is created with its corresponding simple cycle $[u, c_1, \cdots, c_x, v, u]$. Each pair of vertices w_{2i-1} and w_{2i}, $1 \leq i \leq r$, is a Tutte pair. The corresponding σ-vertex for Tutte pair (w_{2i-1}, w_{2i}) is incident on the new π-vertex created. (iii) For each c-vertex y_i, $1 \leq i \leq r$, we create a β-vertex and connect it to the σ-vertex corresponds to Tutte pair (w_{2i-1}, w_{2i}). (iv) For each b-vertex y_i, $1 \leq i \leq r$, the σ-vertex corresponding to the Tutte pair (w_{2i-1}, w_{2i}) is incident on the β-vertex z whose corresponding Tutte component contains w_{2i-1} and w_{2i} if z exists. (v) We merge σ-vertices corresponds to the same Tutte pair and delete degree-1 π-vertices.*

The following lemma can be easily derived from Lemma 2, the definition of Tutte split and the definition of the 3-block graph. An example is shown in Figure ??.

Lemma 4 *Let G' be the graph obtained from G by adding an edge between two demanding vertices u and v, where u and v are in different Tutte components. Given $\{Q_1, \cdots, Q_r\}$, the collection of implied paths between u and v in 3-$blk(G)$, and P_2, the implied path in 2-$blk(G)$ between u and v, the 3-block graph for G' can be obtained from 3-$blk(G)$ by performing the updating operations defined in Definition 6 in $O(|P_2| + \sum_{i=1}^{r} |Q_i|)$ time.* □

We know that if G is biconnected, there can be at most two critical σ-vertices in 3-$blk(G)$ (Claim 3). But this is not true for G is not biconnected (a similar claim is also given in [26]). We now state a claim about the set of all critical σ-vertices in 3-$blk(G)$ for a general graph G. In general, if there are more than 2 critical σ-vertices in 3-$blk(G)$, we can partition the set of critical σ-vertices into at most two subsets such that either a subset has one σ-vertex or Tutte pairs corresponding to critical σ-vertices in a subset share a common cutpoint.

Claim 8 *Let G be a connected graph with $w(G) > 2$. Let $\mathcal{S} = \{s_1, \cdots, s_r\}$ be the set of critical σ-vertices in 3-$blk(G)$. Given $\mathcal{S}$, let $\Im_1 = \{c \mid c$ is a cutpoint that is shared by more than one Tutte pair represented by members of $\mathcal{S}\}$. Let $\Im_2 = \{s \mid s \in \mathcal{S}$ and $\not\exists c \in \Im_1$ such that the Tutte pair represented by s contains $c\}$. Then $|\Im_1| + |\Im_2| \leq 2$.*

<u>Proof:</u> The proof uses 3 propositions and is given in [11]. All propositions are proved by first finding separating degrees of σ-vertices in $\mathcal{S}$ and a lower bound value for $w(G)$ in all possible cases and then deriving a contradiction by the fact that the separating degree of any critical σ-vertex must be equal to $\lceil \frac{w(G)}{2} \rceil$. □

Corollary 2 *Let G be a connected graph with $w(G) > 2$. We can find two demanding vertices u_1 and u_2 in G such that each critical σ-vertex is either in one of the paths in the collection of implied paths between u_1 and u_2 in 3-$blk(G)$ or its corresponding Tutte pair contains a cutpoint in the implied path between them in 2-$blk(G)$.* □

Claim 9 states some conditions under which separating degrees of certain σ-vertices decrease by 1.

Claim 9 *Let G be an undirected graph and let u and v be two demanding vertices in G. Let G' be the graph obtained from G by adding the edge (u,v) and let s be a σ-vertex in 3-$blk(G)$ with $sd(s) \geq 3$. Then $sd(s)$ decreases by 1 in G' if any one of the following conditions is true: (i) s is in one of the path in the collection of implied paths between u and v in 3-$blk(G)$; (ii) the corresponding Tutte pair of s contains a cutpoint in the implied path between u and v in 2-$blk(G)$.* □

Lemma 5 states a condition under which the weight of the graph decreases by 2 after adding an edge.

Lemma 5 *Let G' be the graph obtained from a graph G by adding an edge between two distinct demanding vertices u and v. Then $w(G') = w(G) - 2$ if u and v are in different connected components or in different 2-blocks.* □

By Lemma 5 and Corollary 2, we now show that in any biconnected graph, we can always decrease the lower bound given in Lemma 1 by 1 by adding an edge.

Corollary 3 *Let G be a connected graph that is not biconnected. We can find two demanding vertices u_1 and u_2 in G such that they satisfy the condition given in Corollary 2 and $w(G \cup \{(u_1, u_2)\}) = w(G) - 2$.*

Proof: If we can find two demanding vertices not in the same 2-block that satisfy the condition given in Corollary 2 or any two critical σ-vertices whose Tutte pairs share a common cutpoint, then Lemma 5 shows $w(G \cup \{(u_1, u_2)\}) = w(G) - 2$. Let u_1 and u_2 be in the 2-block H; every critical σ-vertex is also in H and they do not share any cutpoint. ¿From Claim 8, we know that there can be at most two critical σ-vertices whose Tutte pairs do not share cutpoint. If there is only one critical σ-vertex in H, we can easily substitute one of u_1 and u_2 for a demanding vertex in a 2-block that is not H such that u_1 and u_2 satisfy both the conditions given in Corollary 2 and Lemma 5. Thus $w(G \cup \{(u_1, u_2)\}) = w(G) - 2$. For the case of having two critical σ-vertex in H, their degrees in 3-$blk(G)$ must be greater than 2 since G is not biconnected and they are the only two critical σ-vertices in 3-$blk(G)$. By Lemma 3, the number of 3-block leaves decreases by 2. Thus the weight of G decreases by 2 after adding (u_1,u_2). □

5.2 An Algorithm for Triconnecting an Undirected Graph Using the Smallest Number of Edges

In the following algorithm, aug3, we find a pair of demanding vertices u and v in G such that the collection of implied paths $\mathcal{Q}$ in 3-$blk(G)$ between u and v passes through the massive σ-vertex or all critical σ-vertices if G is connected. If G is not connected, u and v are demanding vertices in G that are in different connected components. We also have to make sure that $w(G)$ is reduced by 2 if 3-$blk(G)$ is balanced by satisfying the conditions given in Lemma 5.

{∗ The graph G has at least 4 vertices; the algorithm finds a smallest augmentation to triconnect G. ∗}
graph function aug3(**graph** G);
  T_2 := 2-$blk(G)$; T_3 := 3-$blk(G)$;
  let $w(G)$ be the weight of G; {∗ Definition 3. ∗}
  do $w(G) \geq 2$ **and** ∃ a c-vertex in T_2 →
    if G is not connected →
1.   find y and z that are 3-block leaves or isolated 3-block vertices in different trees of T_3
    | G is connected →
      let d be the largest separating degree among all σ-vertices in T_3;
      if $d > \lceil \frac{w(G)}{2} \rceil$ → {∗ There exists an unique massive σ-vertex in T_3 (Claim 10). ∗}
2.   let s be the massive σ-vertex in T_3;
3.   find two s-chain leaves y and z in T_3
      | $d \leq \lceil \frac{w(G)}{2} \rceil$ →
4.   find y and z that are 3–block leaves or isolated 3-block vertices in T_3 such that demanding vertices u and v of y and z, respectively, satisfy the condition given in Corollary 3 {∗ Corollary 3 shows that this is always possible. ∗}
      fi
    fi;
5.   find demanding vertices u_1 and u_2 of y and z, respectively; {∗ Claim 2 shows that this is always possible. ∗}
    add an edge between u_1 and u_2;
    update the 2-block graph T_2;
    update the 3-block graph T_3;
    if G is connected **and** $d > \lceil \frac{w(G)}{2} \rceil$ →
      $w(G) := w(G) - 1$
    | G is not connected **or** $d \leq \lceil \frac{w(G)}{2} \rceil$ →
      $w(G) := w(G) - 2$
    fi
  od;
  {∗ G is biconnected. ∗}
  return aug2to3(G)
end aug3;

Claim 10 *If there exists a massive σ-vertex s_1 in 3-$blk(G)$, then a σ-vertex s_2 with the largest separating degree among σ-vertices other than s_1 is not massive or critical. Let $\delta = sd(s_1) - \lceil \frac{w(G)}{2} \rceil$. There are at least $2\delta + 2$ s_1-chains in 3-$blk(G)$. Let $\mathcal{M}$ be the set of s_1-chain leaves. By adding $2k$, $k \leq \delta$, edges to connect $2k+1$ vertices of $\mathcal{M}$, we reduce both $sd(s_1)$ and the weight of the graph by k.* □

Claim 11 *Let G be an undirected graph with at least 4 vertices. We can triconnect G by adding $max\{d, \lceil \frac{w(G)}{2} \rceil\}$ edges using algorithm aug3, where d is the largest separating degree among all σ-vertices in 3-$blk(G)$ and $w(G)$ is the weight of G.*

Proof: If G is not connected, algorithm aug3 finds u_1 and u_2 in different connected components at steps 1 and 5. ¿From Lemma 5, we know that $\lceil \frac{w(G)}{2} \rceil$ is decreased by 1. ¿From the definition of separating degree, d is decreased by 1 because the number of connected components is decreased by 1. ¿From Claim 10, we know that there can be at most one massive vertex s if $d > \lceil \frac{w(G)}{2} \rceil$. Algorithm aug3 finds s at step 2. We reduce $max\{d, \lceil \frac{w(G)}{2} \rceil\}$ by 1 by adding a new edge between two s-chain leaves (step 3) if G is connected and unbalanced. If G is balanced, algorithm aug3 adds an edge (u_1,u_2) such that u_1 and u_2 are two demanding vertices satisfy the condition given in Corollary 3. Thus $\lceil \frac{w(G)}{2} \rceil$ is decreased by 1 and G remains balanced. Algorithm aug3 keeps doing this until G is biconnected. Notice that if G is biconnected, $w(G) = l_3(G)$. At this point, algorithm aug3 calls algorithm aug2to3 on the current biconnected graph. Hence by Claim 6, the claim is true. □

Theorem 2 *The triconnectivity augmentation number for a graph G equals $max\{d, \lceil \frac{w(G)}{2} \rceil\}$, where d is the largest separating degree among all σ-vertices in 3-$blk(G)$ and $w(G)$ is the weight of G.* □

5.3 A Linear Time Implementation

Let s_G be the sum of the separating degrees of all Tutte pairs in the 3-block graph of a graph G with n vertices and m edges. By using techniques similar to those described in [10] and the proof of Claim 7, algorithm aug3 can be implemented to run in $O(n+m+s_G)$ time. Since s_G could be $\Omega(n^2)$ for a graph with n vertices, we do not have a linear time implementation for algorithm aug3 if m is not $\Theta(n^2)$. However, Lemma 6 ([26] stated a similar claim which is part of the lemma) tells us of a possible method to obtain a linear time implementation.

Lemma 6 *Given a critical Tutte pair s_1 and another Tutte pair s_2, $s_1 \neq s_2$, in a connected graph G, let s_1 and s_2 share a common cutpoint c. Let $s_1 = (c, a_1)$ and let $s_2 = (c, a_2)$. Then $d_3(s_1) \geq d_3(s_2)+2(d_2(a_2)-1)$. If s_2 is also critical, then $d_2(a_1) = 1$ and $d_2(a_2) = 1$.* □

Lemma 6 says that for a set of more than 1 critical σ-vertex whose corresponding Tutte pairs share a common cutpoint c, the other vertices in these Tutte pairs are not cutpoints. Let $\mathcal{S}_c$ be the set of Tutte pairs share c and let $\mathcal{J}_c$ be the subset of $\mathcal{S}_c$ with the largest degree in 3-$blk(G)$ among all Tutte pairs in $\mathcal{S}_c$. A Tutte pair s in $\mathcal{S}_c$ is a *candidate* if and only if any one of the following conditions is true: (i) $\mathcal{J}_c = \{s\}$; (ii) $s \in \mathcal{J}_c$ and only the vertex c in each Tutte pair in $\mathcal{J}_c$ is a cutpoint; (iii) $s \in \mathcal{J}_c$ and both vertices in s are cutpoints. We can find a candidate for any $\mathcal{S}_c$. Lemma 6 also states that if $|\mathcal{S}_c| \geq 2$ and any one of the candidates is not critical, then none of the Tutte pairs in $\mathcal{S}_c$ is critical.

Let n be the number of vertices and m be the number of edges in the input graph. Using Lemma 6 and the *sorted table* data structure given in [16], we can implement algorithm aug3 in $O(n+m)$ time.

Given a set of p nodes $\mathcal{V}$ with a key in each node whose value is $O(p)$, the *sorted table* is an array $\mathcal{L}$ of $O(p)$ entries. The ith entry of $\mathcal{L}$ points to a doubly linked list which contains all nodes in $\mathcal{V}$ with the key value i. For Tutte pairs that do not share any cutpoint with any other Tutte pairs, we maintain a sorted table using their separating degrees as keys. Entries are updated whenever their separating degrees are changed. The sum of the separating degrees of all σ-vertices in the sorted table is $O(n)$.

Given a sorted table $\mathcal{L}$, an equivalent *sorted list* is a doubly linked list on entries of $\mathcal{L}$ that does not point to an empty list. The relative order of entries in $\mathcal{L}$ is preserved. For each cutpoint c that is shared by Tutte pairs $\mathcal{S}_c = \{(c, a_1), \cdots, (c, a_r)\}$, where $r \geq 2$, we maintain the following information: y_c, a candidate for $\mathcal{S}_c$ (y_c is updated only if the current candidate no longer satisfies the condition of being a candidate) and $\mathcal{Q}_c$, a sorted list on the set of σ-vertices corresponding to Tutte pairs in $\mathcal{S}_c$ with their degrees in the 3-block graph as their key values. Note that we must re-choose y_c, where c is a cutpoint, if and only if a degree in 3-$blk(G)$ for a Tutte pair is changed or a degree in 2-$blk(G)$ for a cutpoint becomes 1. Thus we only have to re-choose $O(n)$ times. Thus the above information can be maintained in totally $O(n)$ time.

For a cutpoint c, let its y_c be (c,a_c). We maintain an entry with key value $sd(y_c)$ in the sorted table if and only if any one of the following conditions is true: (i) a_c is not a cutpoint; (ii) $|\mathcal{S}_{a_c}| \leq 1$; (iii) $|\mathcal{S}_{a_c}| > 1$ and $y_c = y_{a_c}$. This makes sure that if we update (decrease) the degree of a cutpoint, only a constant number of entries in the sorted table are updated. Note that if we have to change y_c for a cutpoint c, we only have to get a constant number of entries in and out the sorted table. Every entry in the sorted table is updated whenever its separating degree is changed.

During the augmentation using algorithm aug3, we create new σ-vertices by creating new polygons. The sum of the degrees in the 3-block graph of all created σ-vertices is $O(n)$. Since the sum of the degrees of all σ-vertices in the 3-block graph, the sum of the degrees of all c-vertices in the 2-block graph and the sum of the separating degree of all σ-vertices in 3-$blk(G)$ are all $O(n)$, the total time to update the sorted table is $O(n)$. Using the above method, we can implement algorithm aug3 in linear time.

Claim 12 *Algorithm aug3 runs in linear time.* □

6 Conclusion

In this paper, we have presented a linear time sequential algorithm for finding a smallest augmentation to triconnect an undirected graph. The algorithm is divided into two stages. During the first stage, we biconnect the input graph. Then we triconnect the resulting biconnected graph using the smallest number of edges in the second stage. We have to make sure that the total number of edges added in these two stages is minimum. We first described a linear time sequential algorithm to triconnect a biconnected graph using the smallest number of edges. Then we presented a linear time algorithm to biconnect a graph such that the set of edges added is a subset of edges needed to triconnect the graph using the smallest number of edges.

References

[1] Guo-Ray Cai & Yu-Geng Sun, " The minimum augmentation of any graph to a k-edge-connected graph," *Networks*, Vol. 19, 1989, pp. 151-172.

[2] Kapali P. Eswaran & R. Endre Tarjan, "Augmentation problems," *SIAM J. Comput.* Vol 5, December 1976, pp. 653-665.

[3] S. Even, *Graph Algorithms,* Computer Science Press, Rockville, MD, 1979.

[4] András Frank, "Augmenting graphs to meet edge-connectivity requirements," *Proc. 31th Annual IEEE Symp. on Foundations of Comp. Sci.*, 1990, pp. 708-718.

[5] H. Frank & W. Chou, "Connectivity considerations in the design of survivable networks," *IEEE Trans. on Circuit Theory,* Vol. CT-17, November 1970, pp. 486-490.

[6] Greg N. Frederickson & Joseph Ja'Ja', "Approximation algorithms for several graph augmentation problems," *SIAM J. Comput.*, Vol. 10, May 1981, pp. 270-283.

[7] D. Gusfield, "Optimal mixed graph augmentation," *SIAM J. Comput.*, Vol. 16, August 1987, pp. 599-612.

[8] D. Harel & R. E. Tarjan, "Fast algorithms for finding nearest common ancestors," *SIAM J. Comput.*, Vol. 13, 1984, pp. 338-355.

[9] J. E. Hopcroft & R. E. Tarjan, "Dividing a graph into triconnected components," *SIAM J. Comput.*, Vol. 2, September 1973, pp.135-158.

[10] Tsan-sheng Hsu & Vijaya Ramachandran, "On finding a smallest augmentation to biconnect a graph," Tech. Rep. TR-91-12, U.T. Austin, 1991.

[11] Tsan-sheng Hsu & Vijaya Ramachandran, "A linear time algorithm for triconnectivity augmentation," Manuscript, U.T. Austin, 1991.

[12] Yoji Kajitani & Shuichi Ueno, "The minimum augmentation of a directed tree to a k-edge-connected directed graph," *Networks*, Vol. 16, 1986, pp. 181-197.

[13] Arkady Kanevsky, "Vertex connectivity of graphs: algorithms and bounds," Tech. Rep. ACT-97 (Ph.D. thesis), Coordinated Science lab., Univ. of Illinois, Urbana, 1988.

[14] D. Naor, D. Gusfield & C. Martel, "A fast algorithm for optimally increasing the edge-connectivity," *Proc. 31th Annual IEEE Symp. on Foundations of Comp. Sci.*, 1990, pp. 698-707.

[15] Vijaya Ramachandran, "Parallel open ear decomposition with applications to graph biconnectivity and triconnectivity," invited chapter for *Synthesis of Parallel Algorithms*, J. H. Reif, editor, Morgan-Kaufmann.

[16] Arnie Rosenthal & Anita Goldner, "Smallest augmentations to biconnect a graph," *SIAM J. Comput.*, Vol. 6, March 1977, pp. 55-66.

[17] Danny Soroker, "Fast parallel strong orientation of mixed graphs and related augmentation problems," *Journal of Algorithms*, 9, 1988, pp. 205-223.

[18] Kenneth Steiglitz, Peter Weiner & D. J. Kleitman, "The design of minimum-cost survivable networks," *IEEE Trans. on Circuit Theory*, Vol. CT-16, November 1969, pp. 455-460.

[19] R. E. Tarjan, *Data Structures and Network Algorithms*, SIAM Press, Philadelphia, PA, 1983.

[20] W. T. Tutte, *Connectivity in Graphs*, University of Toronto Press, 1966.

[21] Shuichi Ueno, Yoji Kajitani & Hajime Wada, "Minimum augmentation of a tree to a k-edge-connected graph," *Networks*, Vol. 18, 1988, pp 19-25.

[22] T. Watanabe, "An efficient way for edge-connectivity augmentation," Tech. Rep. ACT-76, Coordinated Science lab., Univ. of Illinois, Urbana, 1987.

[23] T. Watanabe, Y. Higashi & A. Nakamura, "Graph augmentation problems for a specified set of vertices," *Proc. first Ann. Intl. Sym. on Algorithms*, 1990, pp. 378-387.

[24] T. Watanabe & A. Nakamura, "On a smallest augmentation to triconnect a graph," Tech. Rep. C-18, Department of Applied Mathematics, Hiroshima University, Japan (1983, revised 1987).

[25] T. Watanabe & A. Nakamura, "Edge-connectivity augmentation problems," *J. Comput. System Sci.*, 35(1987), pp. 96-144.

[26] T. Watanabe & A. Nakamura, "3-Connectivity augmentation problems," *Proc. 1988 IEEE Intl. Sym. on Circuits and Systems*, 1988, pp. 1847-1850.

[27] T. Watanabe, T. Narita & A. Nakamura, "3-Edge-Connectivity augmentation problems," *Proc. 1989 IEEE Intl. Sym. on Circuits and Systems*, 1989, pp.335-338.